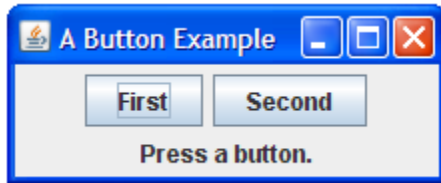


## Εργαστήριο 1 - 1<sup>η</sup> Άσκηση - Ανάλυση

**Εκφώνηση:** Δημιουργείστε εφαρμογή σε Java Swing με χρήση του IDE NetBeans όπου θα παρουσιάζεται ποιο κουμπί πατήθηκε. Η εφαρμογή θα μοιάζει ως εξής:

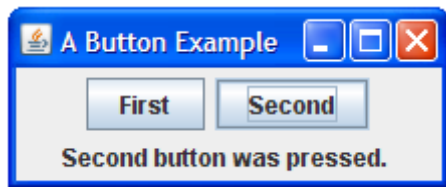
Πρώτο Βήμα: Αρχική Εικόνα



Δεύτερο Βήμα: Πατήθηκε το πρώτο κουμπί

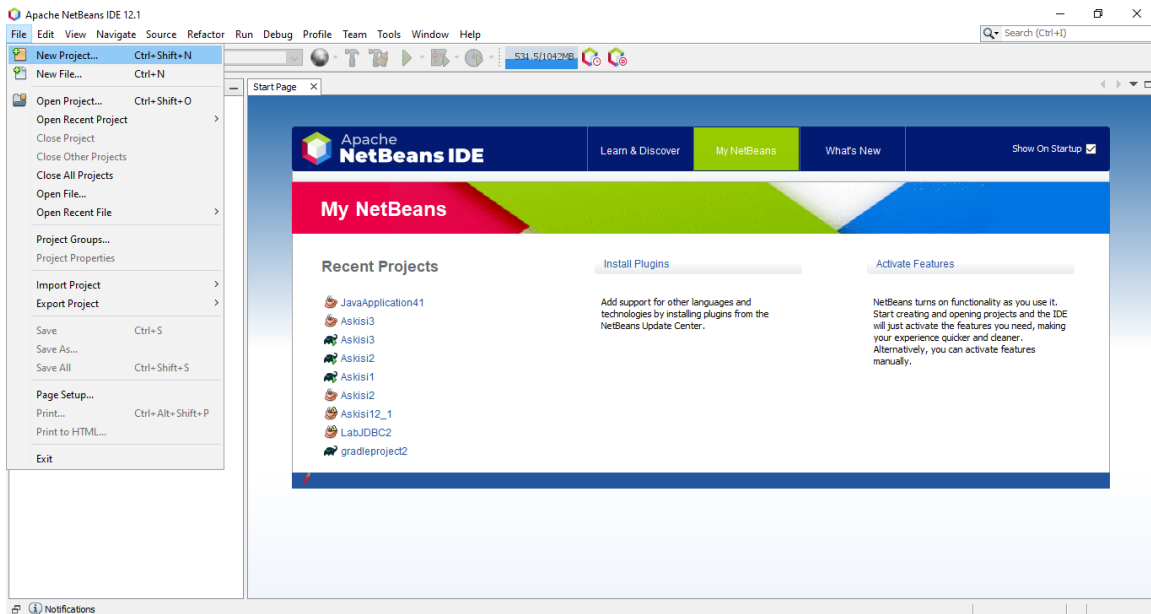


Τρίτο Βήμα: Πατήθηκε το δεύτερο κουμπί

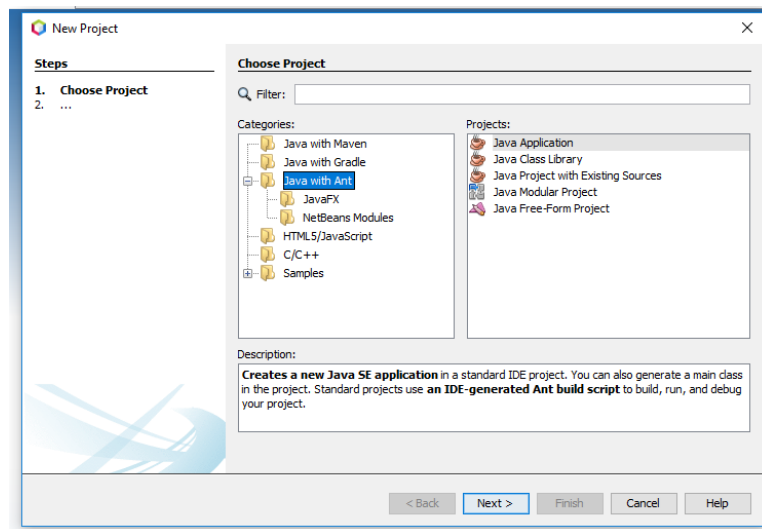


## Ανάλυση και εξήγηση λύσης

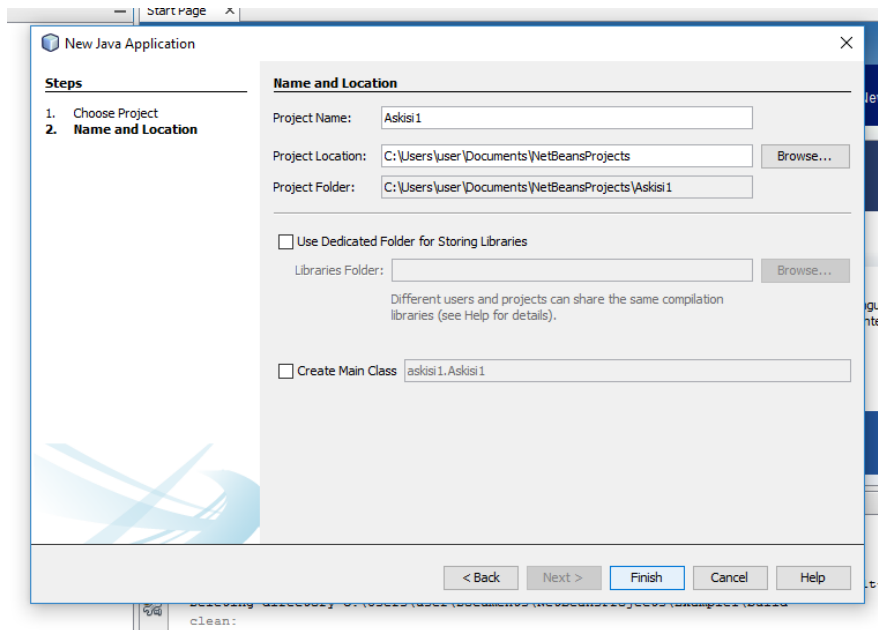
**Βήμα 1:** Δημιουργούμε ένα νέο project στο NetBeans με όνομα Askisi1.



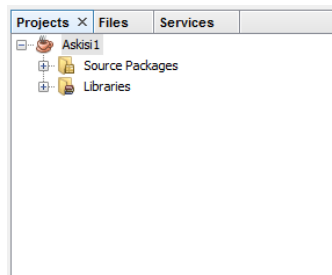
Εικόνα 1: Δημιουργία νέου Project



Εικόνα 2: Ο τύπος του project θα είναι Java Application

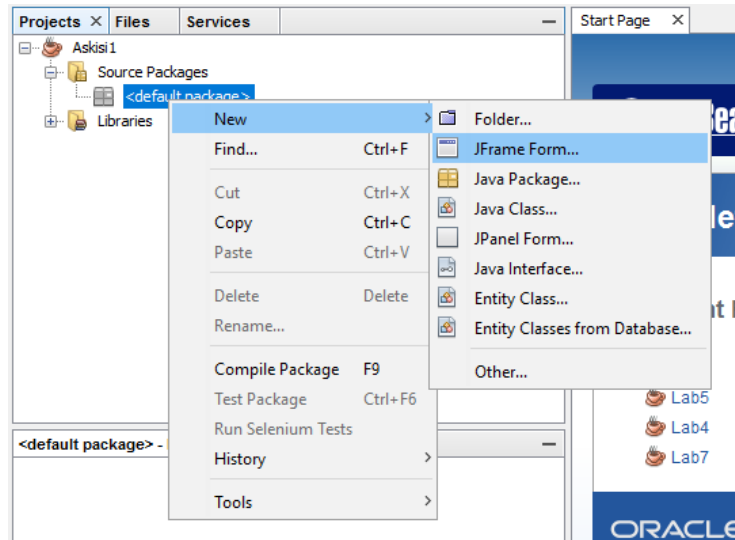


Εικόνα 3: Ονομάζουμε το project Askisi1

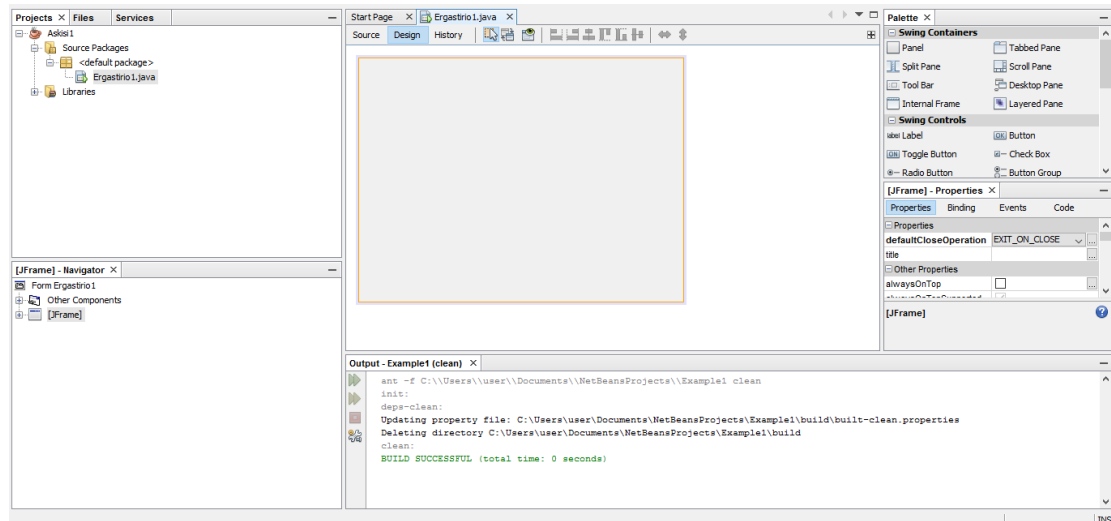


Εικόνα 4: Το project δημιουργήθηκε

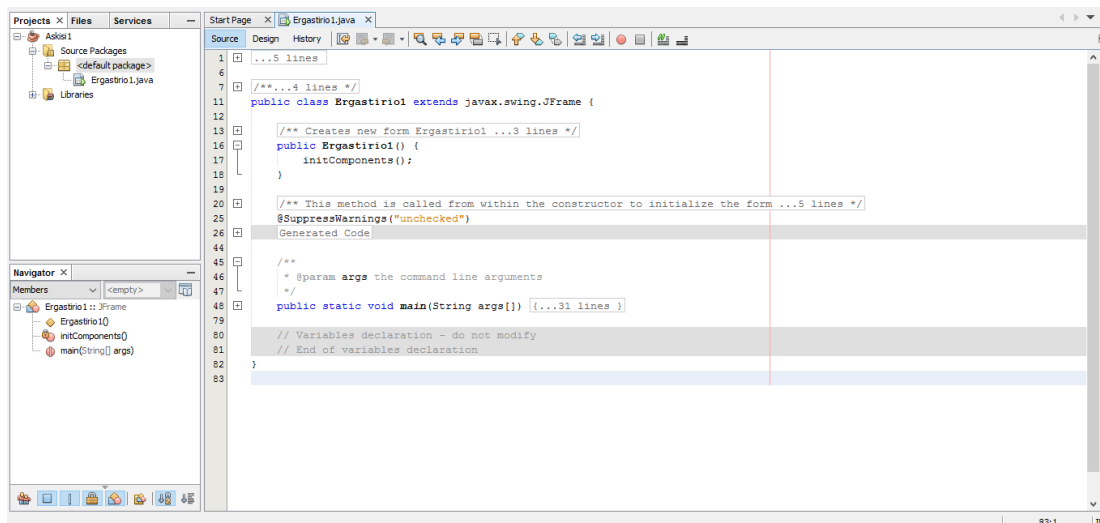
**Βήμα 2:** Δημιουργούμε ένα νέο Java αρχείο τύπου JFrame Form με όνομα Ergastirio1.



Εικόνα 5: Πατώντας το δεξί πλήκτρο του ποντικιού πάνω στο package.



Εικόνα 6: Το αρχείο δημιουργήθηκε. Στην εικόνα βλέπουμε το Design Mode.



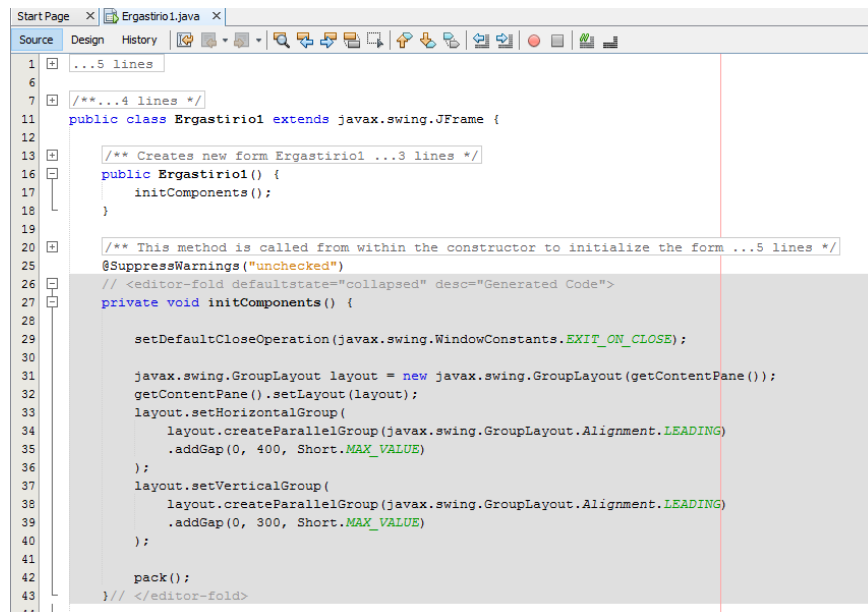
Εικόνα 7: To Source Mode. Ο κώδικας της εφαρμογής

Παρατηρούμε ότι το πρόγραμμα έχει τα παρακάτω βασικά κομμάτια.

- `public class Ergastirio1 extends javax.swing.JFrame {`  
 Η κλάση του java αρχείου. Κάνει extends το package javax.swing.JFrame. Αυτό το πακέτο περιέχει όλες τις κλάσεις και μεθόδους για τη διαχείριση ενός εξωτερικού υποδοχέα (παραθύρου) στη Java Swing.
- `public Ergastirio1().`  
 Ο δομητής της κλάσης. Είναι η πρώτη μέθοδος που εκτελείται όταν τρέχει η εφαρμογή. Επομένως, στο σώμα αυτής της μεθόδου τοποθετούμε τις αρχικές τιμές που θέλουμε να έχει η εφαρμογή. Στην Εικόνα 7 φαίνεται ότι η μέθοδος περιέχει μόνο μία κλήση μεθόδου την `initComponents()`. Η `initComponents()` σχεδιάζει την αρχική εικόνα της διεπαφής. Ότι έχουμε βάλει στον καμβά της διεπαφής και τις αρχικές ιδιότητες που έχουμε ορίσει. Στην Εικόνα 8, φαίνεται ο κώδικας της `initComponents`, όπως δημιουργήθηκε αυτόματα δημιουργώντας το JFrame.
- `public static void main(String args[])`  
 Τα προγράμματα Swing είναι event-driven (προγραμματισμός γεγονότων) γι' αυτό και ενεργοποιούνται στο νήμα-γεγονότων και όχι στο νήμα-main. Με το αντικείμενο Runnable ενεργοποιούμε το νήμα-γεγονότων. Τις διεπιφάνειες τις ενεργοποιούμε πάνω στο νήμα-γεγονότων με τις μεθόδους
  1. `static invokeLater(Runnable obj)` και
  2. `static invokeAndWait(Runnable obj)`

Συνήθως προτιμούμε την πρώτη για desktop εφαρμογές ενώ τη δεύτερη για Web εφαρμογές. Στην Εικόνα 9, βλέπουμε τον αυτόματο κώδικα που

δημιουργήθηκε για το πρώτο μας πρόγραμμα. Το νήμα δημιουργείται στη μέθοδο `run()` η οποία εμφανίζει τον καμβά της εφαρμογής με την εντολή `new Ergastirio1().setVisible(true)`, όπως έχει σχεδιασθεί με την `initComponents()`.

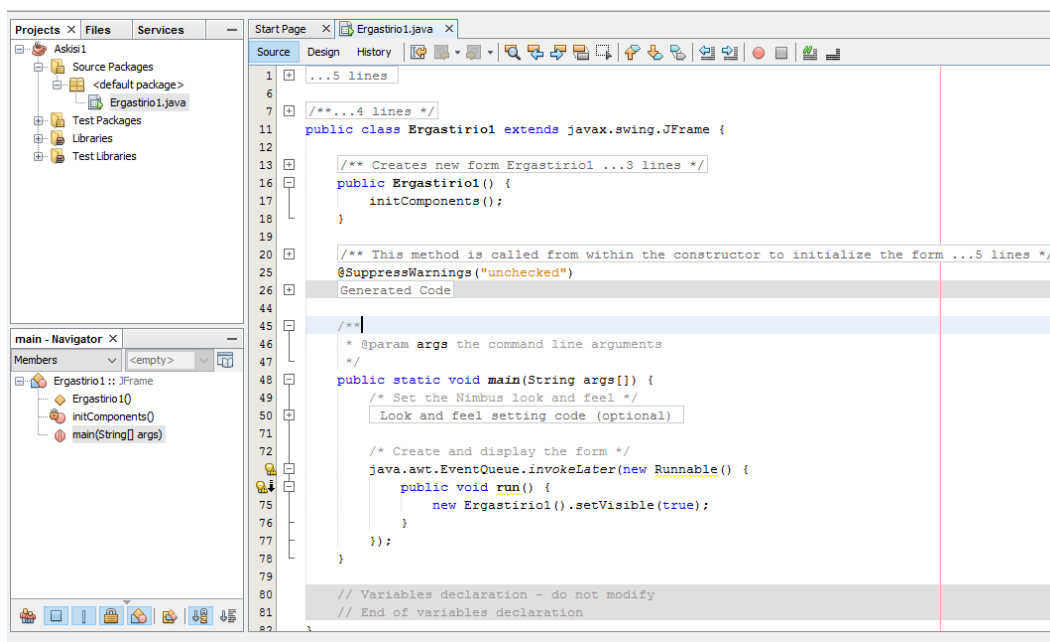


```

1  ...5 lines
6
7  /**...4 lines */
11 public class Ergastirio1 extends javax.swing.JFrame {
12
13     /** Creates new form Ergastirio1 ...3 lines */
14     public Ergastirio1() {
15         initComponents();
16     }
17
18     /** This method is called from within the constructor to initialize the form ...5 lines */
19     @SuppressWarnings("unchecked")
20     // <editor-fold defaultstate="collapsed" desc="Generated Code">
21     private void initComponents() {
22
23         setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
24
25         javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
26         getContentPane().setLayout(layout);
27         layout.setHorizontalGroup(
28             layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
29                 .addGroup(layout.createSequentialGroup()
30                     .addGap(0, 400, Short.MAX_VALUE)
31                     .addContainerGap())
32             .addGroup(layout.createSequentialGroup()
33                 .addGap(0, 300, Short.MAX_VALUE)
34                 .addContainerGap())
35         );
36         pack();
37     }
38     // </editor-fold>
39
40
41
42
43
44

```

Εικόνα 8: Η μέθοδος initComponents



```

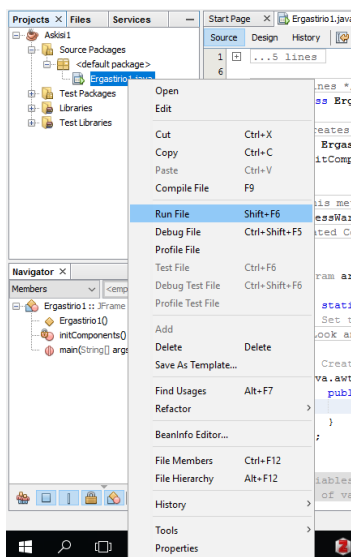
1  ...5 lines
6
7  /**...4 lines */
11 public class Ergastirio1 extends javax.swing.JFrame {
12
13     /** Creates new form Ergastirio1 ...3 lines */
14     public Ergastirio1() {
15         initComponents();
16     }
17
18     /** This method is called from within the constructor to initialize the form ...5 lines */
19     @SuppressWarnings("unchecked")
20     // <editor-fold defaultstate="collapsed" desc="Generated Code">
21     private void initComponents() {
22
23         setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
24
25         javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
26         getContentPane().setLayout(layout);
27         layout.setHorizontalGroup(
28             layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
29                 .addGroup(layout.createSequentialGroup()
30                     .addGap(0, 400, Short.MAX_VALUE)
31                     .addContainerGap())
32             .addGroup(layout.createSequentialGroup()
33                 .addGap(0, 300, Short.MAX_VALUE)
34                 .addContainerGap())
35         );
36         pack();
37     }
38     // </editor-fold>
39
40
41
42
43
44
45
46     /**
47      * @param args the command line arguments
48      */
49     public static void main(String args[]) {
50         // Set the Nimbus look and feel
51         // Look and feel setting code (optional)
52
53         /* Create and display the form */
54         java.awt.EventQueue.invokeLater(new Runnable() {
55             public void run() {
56                 new Ergastirio1().setVisible(true);
57             }
58         });
59     }
60
61     // Variables declaration - do not modify
62     // End of variables declaration
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82

```

Εικόνα 9: Η μέθοδος main

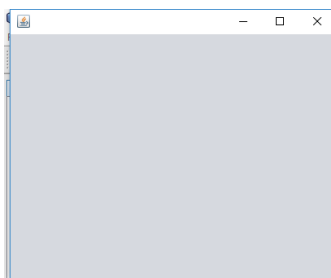
**Βήμα 3:** Εκτέλεση του προγράμματος. Η εκτέλεση του προγράμματος γίνεται με πολλούς τρόπους μέσα από το NetBeans. Ένας από αυτούς, όπως φαίνεται στην Εικόνα 10, γίνεται αν πατήσουμε το δεξί πλήκτρο του ποντικιού πάνω στο αρχείο Java, το `Ergastirio1.java`

και επιλέξουμε από το μενού την εντολή *Run File*. Η εκτέλεση του προγράμματος, όπως βλέπουμε στο μενού, μπορεί να γίνει και με το συνδυασμό πλήκτρων *Shift+F6*.



Εικόνα 10: Εκτέλεση προγράμματος

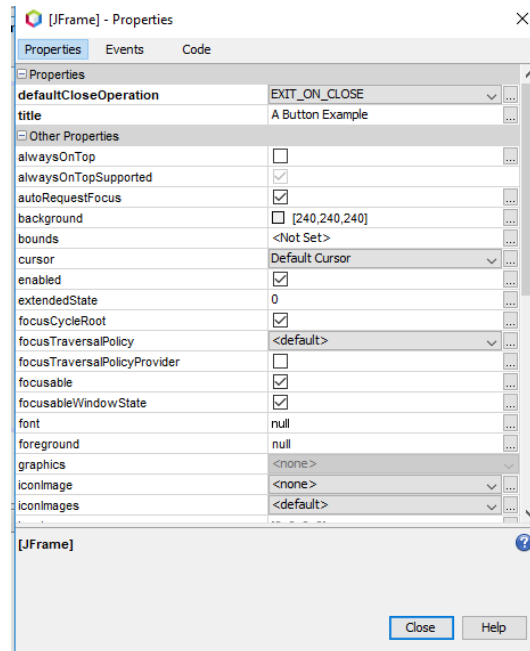
Εκτελούμε το πρόγραμμα και βλέπουμε το αποτέλεσμα της Εικόνα 11. Ένα κλασικό παράθυρο.



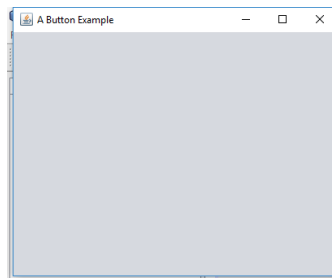
Εικόνα 11: Η εκτέλεση του προγράμματος

**Βήμα 4:** Ορισμός Ιδιοτήτων στο JFrame. Παρατηρούμε ότι δεν έχει τίτλο, ενώ στην εκφώνηση της άσκησης το παράθυρο έχει τον τίτλο «A Button Example». Είναι πολύ εύκολο να ορίσουμε τον τίτλο ενός παραθύρου. Ο καμβάς, δηλαδή το αντικείμενο JFrame, έχει ιδιότητες. Μία από αυτές είναι ο τίτλος. Για να ορίσουμε τις ιδιότητες που θέλουμε να έχει ο καμβάς φροντίζουμε να είμαστε σε Design Mode. Πατάμε το δεξί πλήκτρο του ποντικιού πάνω στον καμβά (το γκρι παραλληλόγραμμο) ή πάνω στο αντικείμενο JFrame (στον Navigator, στο κάτω αριστερό παράθυρο του NetBeans) και επιλέγουμε *Properties/Ιδιότητες*. Στην Εικόνα 12, παρουσιάζονται οι ιδιότητες του JFrame. Η δεύτερη ιδιότητα είναι ο τίτλος του παραθύρου. Συμπληρώνουμε εκεί τον επιθυμητό τίτλο (π.χ. A Button Example) που θέλουμε να έχει το παράθυρο της εφαρμογής μας όταν τρέχει και το ξαναεκτελούμε. Το αποτέλεσμα της εκτέλεσης φαίνεται στην Εικόνα 13. Όποτε ορίζουμε ή τροποποιούμε μία ιδιότητα αλλάζει και ο κώδικας

της μεθόδου  `initComponents` . Στην Εικόνα 14, βλέπουμε στο  `source code`  της εφαρμογής τη γραμμή 30. Η συγκεκριμένη εντολή προστέθηκε όταν ορίσαμε τον τίτλο του παραθύρου.



Εικόνα 12: Οι ιδιότητες του  `JFrame` , του καμβά της διεπαφής



Εικόνα 13: Το παράθυρο της διεπαφής έχει και τίτλο



```

19
20
21
22
23
24
25 /** This method is called from within the constructor to initialize the form ...5 lines */
26 @SuppressWarnings("unchecked")
27 // <editor-fold defaultstate="collapsed" desc="Generated Code">
28 private void initComponents() {
29
30     setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
31     setTitle("A Button Example");
32
33     javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
34     getContentPane().setLayout(layout);
35     layout.setHorizontalGroup(
36         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
37             .addGap(0, 400, Short.MAX_VALUE)
38     );
39     layout.setVerticalGroup(
40         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
41             .addGap(0, 300, Short.MAX_VALUE)
42     );
43     pack();
44 } // </editor-fold>
45

```

Εικόνα 14: Source Code – γραμμή 30 – προστέθηκε η εντολή setTitle

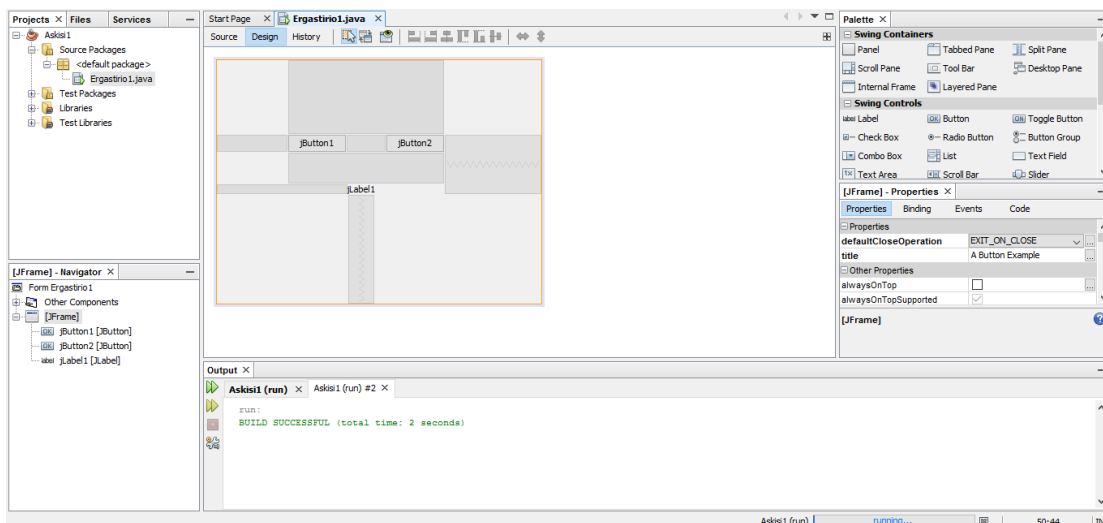
Το JFrame, όπως και τα περισσότερα αντικείμενα ενός καμβά έχουν πολλές ιδιότητες. Μία άλλη ιδιότητα που παρουσιάζει ενδιαφέρον είναι η πρώτη ιδιότητα στο παράθυρο. Με αυτήν την ιδιότητα ορίζεται η συμπεριφορά που θα έχει το πρόγραμμα όταν κλείσει η διεπαφή. Όπως έχουμε αναφέρει και νωρίτερα η γραφική διεπαφή είναι ανεξάρτητη από τη λειτουργικότητα της εφαρμογής (business logic). Ουσιαστικά τρέχουν παράλληλα και όταν κλείσουμε τη γραφική διεπαφή θα πρέπει να ορίσουμε αν θα κλείσει το άλλο κομμάτι της εφαρμογής ή αν θα συνεχίσει να τρέχει στο παρασκήνιο (στη μνήμη). Τέσσερις είναι οι επιλογές της συγκεκριμένης ιδιότητας:

- EXIT\_ON\_CLOSE : στην έξοδο σταματάει και η εφαρμογή.
- DISPOSE: στην έξοδο σταματάει και η εφαρμογή.
- HIDE: στην έξοδο η εφαρμογή δουλεύει στο φόντο.
- DO\_NOTHING: στην έξοδο να μην γίνει τίποτα.

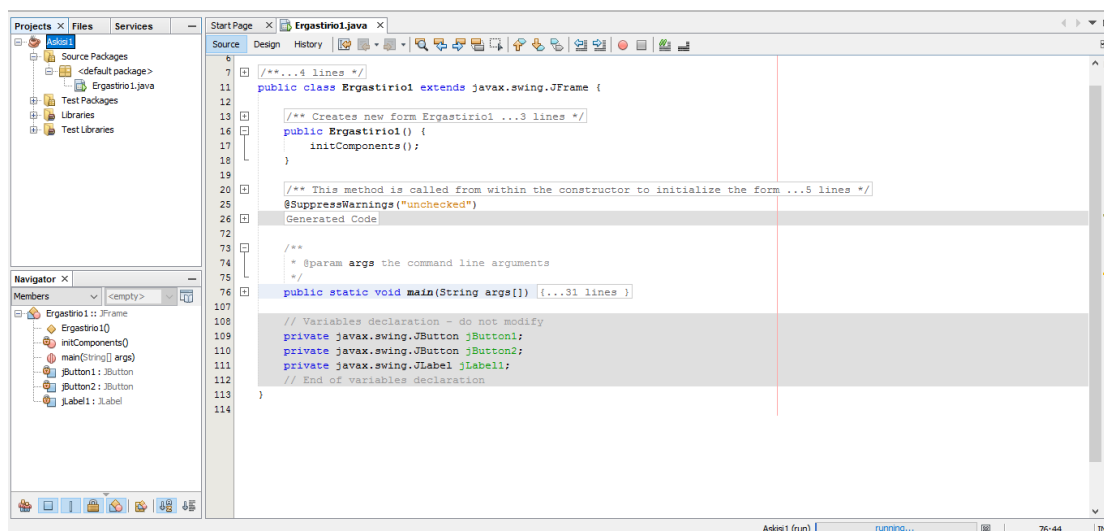
Η συγκεκριμένη ιδιότητα, στο source code, αντιστοιχεί στη γραμμή 29.

**Βήμα 5:** πρόσθεση αντικειμένων στον καμβά και μορφοποίησή τους. Στην εκφώνηση της άσκησης βλέπουμε ότι στο παράθυρο (στον καμβά) υπάρχουν δύο κουμπιά (JButton) και μία ετικέτα (JLabel). Για να προσθέσουμε αυτά τα αντικείμενα στον καμβά επιλέγουμε από την εργαλειοθήκη (πάνω δεξί παράθυρο στο NetBeans – Swing Controls) ένα αντικείμενο και το «σέρνουμε» στο επιθυμητό σημείο πάνω στον καμβά. Στην άσκηση μας, «σέρνουμε» δύο Button και ένα Label και τα τοποθετούμε στην κατάλληλη θέση όπως φαίνεται στην Εικόνα 15. Παρατηρούμε στο Navigator του JFrame, ότι τα αντικείμενα αυτά έχουν τοποθετηθεί κάτω από το JFrame, ορίζοντας ένα δέντρο. Στο Navigator, το στοιχείο Other Components περιέχει τα αόρατα αντικείμενα μίας διεπαφής κατά την εκκίνηση. Αυτά μπορεί να είναι ολόκληρα παράθυρα, τα οποία θα τα φέρουμε στην επιφάνεια την κατάλληλη στιγμή, συνήθως μετά την ενεργοποίηση ενός γεγονότος. Επίσης, στο Source Code στη μέθοδο initComponents παρατηρούμε ότι έχουν προστεθεί αρκετές εντολές αυτόματα, μόλις προσθέσαμε τα τρία αντικείμενα. Τέλος, στην Εικόνα 16 παρατηρούμε ότι στο τέλος του κώδικα στο Source Code

προστέθηκαν τρεις γραμμές κώδικα (γραμμές 109-111) για τη δήλωση των αντικειμένων που προσθέσαμε στον Καμβά, στο τμήμα δηλώσεων (Variable Declaration).



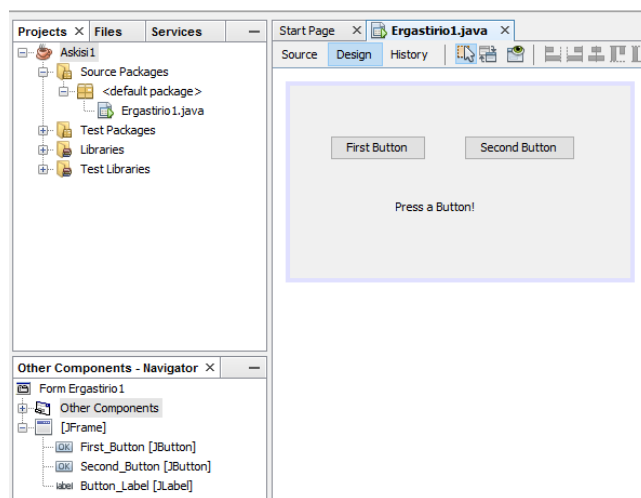
Εικόνα 15: Στον καμβά περιέχονται τρία αντικείμενα



Εικόνα 16: Για κάθε αντικείμενο που προστέθηκε στον καμβά δηλώθηκε κατάλληλα μία μεταβλητή

Το πρώτο πράγμα που πρέπει να κάνουμε για κάθε αντικείμενο είναι να ορίσουμε το όνομα με το οποίο θα αναφερόμαστε σε αυτό, δηλαδή το όνομα της μεταβλητής και φυσικά το κείμενο ή την εικόνα που θα παρουσιάζει στο χρήστη. Αυτά μπορούμε να τα ορίσουμε με χρήση του ειδικού μενού για κάθε αντικείμενο. Έτσι, πατάμε το δεξί πλήκτρο του ποντικιού πάνω στο JButton1 και επιλέγουμε την πρώτη επιλογή του μενού «Edit Text» για να αλλάξουμε το κείμενο που περιέχει το κουμπί – αλλάζτε το κείμενο σε «First Button» ή «Πρώτο Κουμπί». Στη συνέχεια επιλέγουμε τη δεύτερη επιλογή του μενού «Change Variable Name» και ορίζουμε το όνομα της μεταβλητής. Ως όνομα μεταβλητής δίνουμε ένα περιγραφικό όνομα που ξεκινάει από γράμμα (μόνο λατινικούς χαρακτήρες) και δεν έχει κενά – αλλάζτε το όνομα της

μεταβλητής σε `First_Button`. Κάντε τις αντίστοιχες αλλαγές και στο `JButton2` και στο `JLabel1`. Στην Εικόνα 17, παρατηρούμε το αποτέλεσμα στην εμφάνιση των αντικειμένων και τα ονόματα των μεταβλητών τους στο παράθυρο του Navigator.

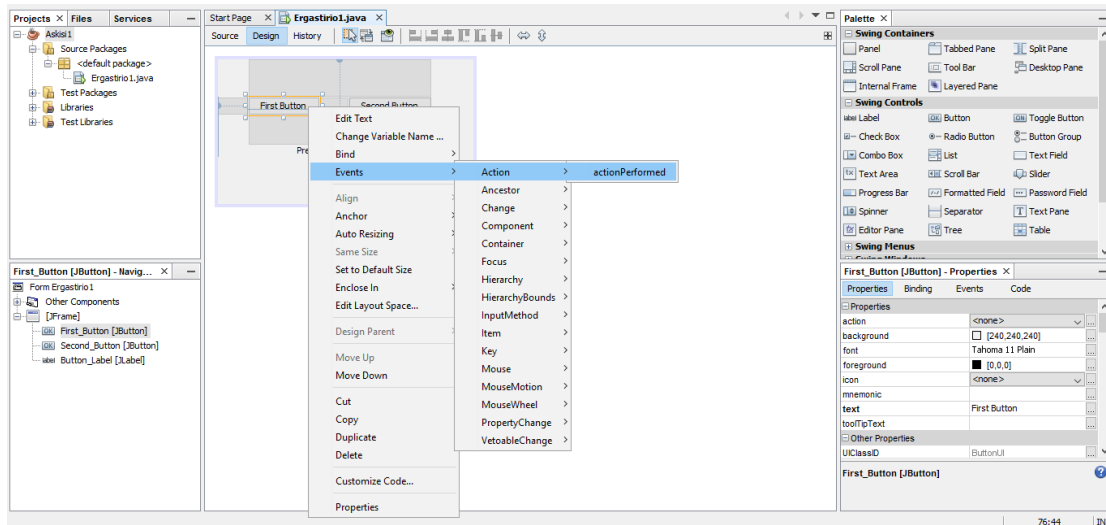


Εικόνα 17: Η εικόνα της εφαρμογής

Εκτελούμε το πρόγραμμα. Παρατηρούμε ότι όταν πατάμε τα κουμπιά δεν γίνεται τίποτα. Στο επόμενο βήμα θα πρέπει να δώσουμε «ζωή» στο πρόγραμμά μας. Να προγραμματίσουμε τη διάδραση, όπου όταν ο χρήστης κάνει μία ενέργεια τότε το πρόγραμμα θα απαντά.

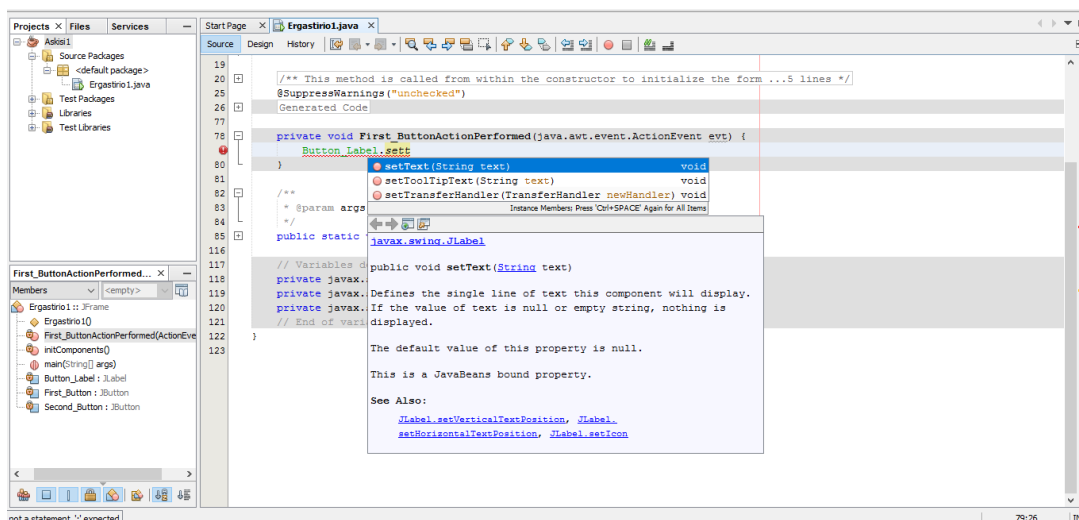
**Βήμα 6:** Μοντέλο αποστολής γεγονότων. Στόχος της άσκησης είναι όταν ο χρήστης πατήσει με το αριστερό πλήκτρο του ποντικιού πάνω στο κουμπί *First\_Button* να αλλάξει το περιεχόμενο της ετικέτας σε «First Button was Pressed». Η λογική του μοντέλου αποστολής γεγονότων είναι όταν συμβεί ένα γεγονός σε ένα αντικείμενο αμέσως να εκτελεστεί μία ενέργεια (μέθοδος). Σε κάθε αντικείμενο μπορούμε να ορίσουμε πολλά γεγονότα τα οποία θα εκτελούν διαφορετικές ενέργειες. Αυτό υλοποιείται με πολύ απλά βήματα.

1. Πατάμε το δεξί πλήκτρο του ποντικιού πάνω στο αντικείμενο που θέλουμε να ορίσουμε ένα γεγονός/event και επιλέγουμε από το μενού την επιλογή Events. Μας παρουσιάζεται ένα υπομενού, διαφορετικό για κάθε αντικείμενο με όλα τα δυνατά γεγονότα που μπορεί να εφαρμοστούν σε αυτό το αντικείμενο. Για το παράδειγμα μας επιλέγουμε το γεγονός *Action* και τη μέθοδο *actionPerformed*, όπως φαίνεται στην Εικόνα 18.

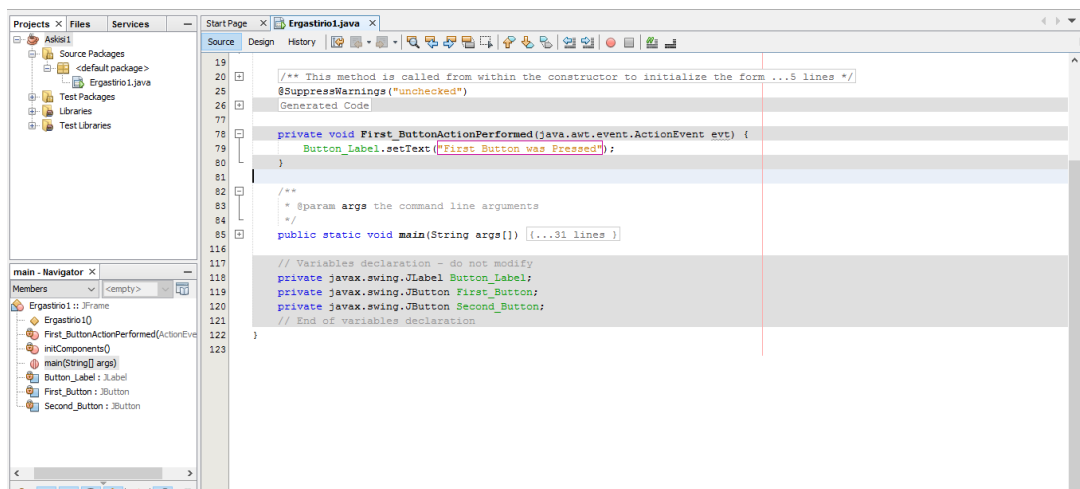


Εικόνα 18: Ορισμός γεγονότος

2. Οδηγούμαστε στο Source Code, όπου δημιουργήθηκε αυτόματα μία μέθοδος, η `First_ButtonActionPerformed`. Προσοχή, δεν μπορούμε να γράψουμε στις «γκρι» περιοχές στο NetBeans. Με γκρι σημειώνεται ότι δημιουργείται αυτόματα από το πρόγραμμα. Στο σώμα της μεθόδου προσθέτουμε τις κατάλληλες εντολές που θέλουμε να εκτελούνται όταν «πυροδοτείται» το γεγονός. Για να αλλάξει το περιεχόμενο της ετικέτας χρησιμοποιούμε τη μέθοδο `setText()`. Έτσι, όπως φαίνεται στην Εικόνα 19, στο σώμα της μεθόδου γράφουμε τη μεταβλητή που αναφέρεται στην ετικέτα μία τελεία και το όνομα της μεθόδου. Μόλις γράψουμε την τελεία ανοίγει αυτόματα ένα μενού που μας προτείνει όλες τις μεθόδους που μπορούμε να εφαρμόσουμε στο συγκεκριμένο τύπου αντικειμένου και φιλτράρει τις μεθόδους σύμφωνα με τα γράμματα που πληκτρολογούμε ενώ ταυτόχρονα μας παρουσιάζει και το signature της επιλεγμένης μεθόδου. Επιλέξτε την `setText()` και συμπληρώστε το επιθυμητό κείμενο όπως φαίνεται στην Εικόνα 20. Εκτελέστε το πρόγραμμα και θα δείτε ότι όποτε πατάτε το κουμπί `First_Button` θα αλλάζει κατάλληλα το κείμενο στην ετικέτα.

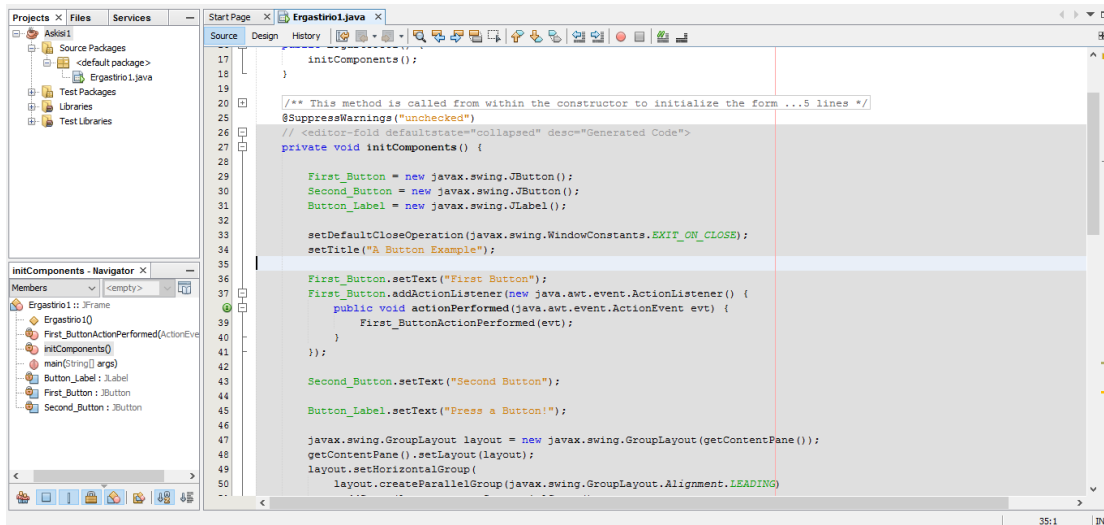


Εικόνα 19: Εισαγωγή κειμένου στην μέθοδο



Εικόνα 20: Ολοκληρώθηκε η μέθοδος

Ένα σημείο πολύ σημαντικό στη λειτουργία του μοντέλου αποστολής γεγονότων είναι ο Ακροατής/Listener. Ο ακροατής είναι υπεύθυνος να παρακολουθεί τότε θα συμβεί το γεγονός. Επιλέγοντας τη δημιουργία ενός γεγονότος/event για ένα αντικείμενο εκτός από τη μέθοδο που περιγράψαμε παραπάνω προστίθεται στον κώδικα και ο ακροατής στη μέθοδο *initComponents()*. Έτσι, στο παράδειγμα μας προστέθηκαν οι γραμμές 37-41 (Εικόνα 21), όπου στην γραμμή 37 δηλώνεται ο ακροατής στο αντικείμενο *First\_Button* όταν πατηθεί το αριστερό πλήκτρο του ποντικιού (action) πάνω στο αντικείμενο. Όταν συμβεί αυτό το γεγονός, ο ακροατής καλεί τη μέθοδο που θα εκτελέσει την ενέργεια – γραμμή 39.



Εικόνα 21: Ο ακροατής του γεγονότος

Για να ολοκληρωθεί η άσκηση προσθέστε το αντίστοιχο γεγονός και στο `Second_Button`.